Data Structures (CSC 212)

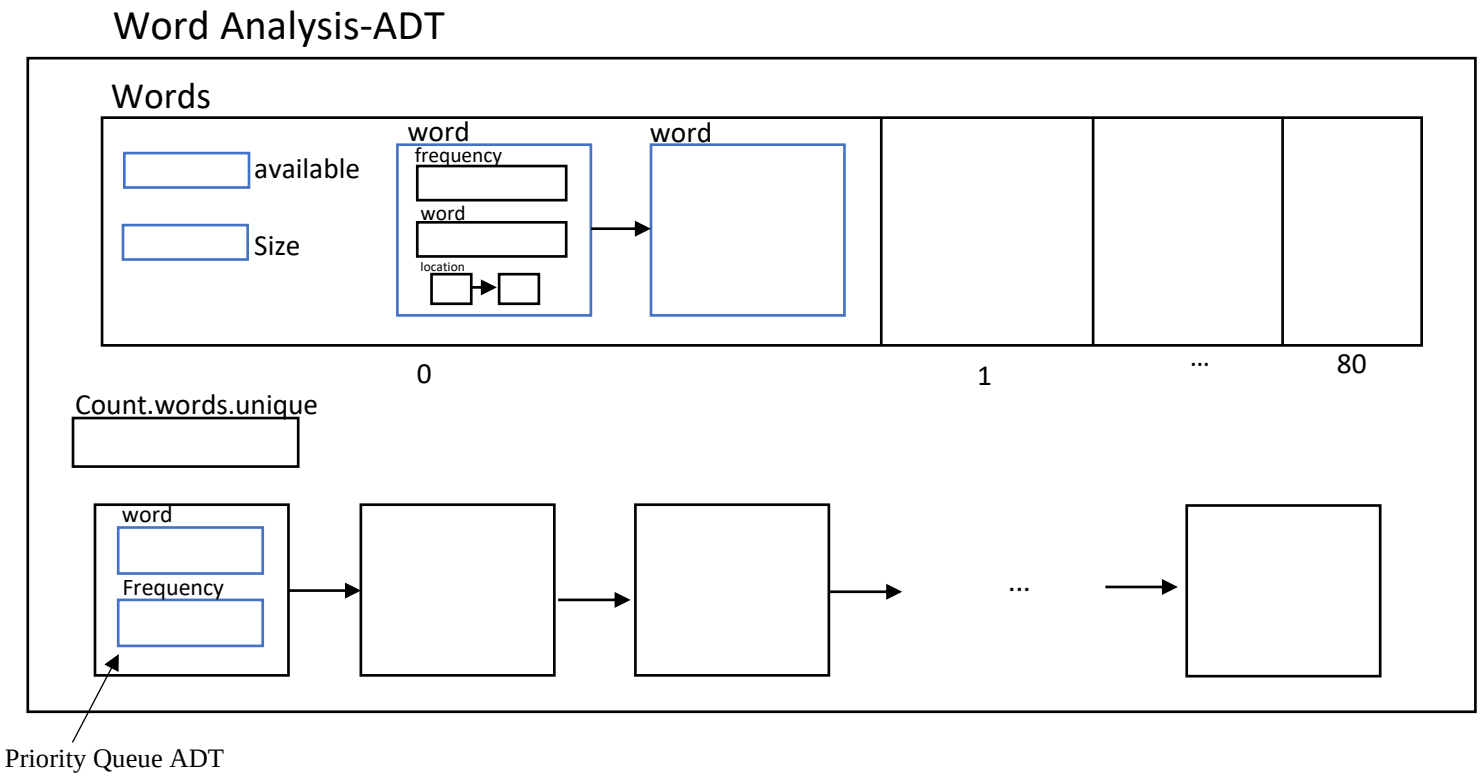
Course Project (Phase1)

GROUP 1

Student name	Student ID	Section number
Maysam Alzahrani(Leader)	442200873	41198
Noura alruwais	442200135	41198

Instructor:Lubna Alhinti

(a) Give a graphical representation of the ADT to show its structure. Make sure to label the diagram clearly.



(b) Write at least one paragraph describing your diagram from part (a). Make sure to clearly explain each component in your design. Also, discuss and justify the choices and the assumptions you make.

We have bulid array with maximum length of words as it size, within each cell there is Linked-list and all words with the same size of array index, so that we can use length of the word to search for it or save in the linkedlist. For Every word has Linked-List with the locations where it appears in the file using line number, word number, and frequency of it; Then Reading from file, using length of the word go direct for room with same length as it index check availability of the word in the list if it is available just increase the frequency and update locations of that word, and if not just add it as new word with frequency=1 We utilize Priority Queue ADT to order all words using frequency as the priority variable by copying all items linked lists of all array which will be sorted Descending order.

(c) Give a specification of the operations (1), (2), (3), (4), (5), (6), and (7) as well as any other supporting operations you may need to read the text from a text file and store the results in the ADT (e.g., insert).

Method name	requires	input	result	output
Readfile (String filename, Linked List words)	file is not empty	filename	adding data from file and store it in the Array with words linkedlist which depend on the length of words by adding new node with word, frequency and it is location in the statement, if it added before just update frequency and add new location in the locations linked list, update the counter for unique words for every inserting new word. after finishing adding all words, sort all item based on frequency numbers store it in the main memory, using priority queue and frequency value as priority value, keeping all sorted words and their frequencies in PQ linked list. keep in mind when reading from file that words are separated by at least one space, Single letter words are counted as words, Punctuation is to be ignored, Hyphenated words or apostrophized words are to be read as single words.	NONE
fileLength (integer size)	Array words is not empty	NONE	Return the total number of words in a text file without duplication count of words	size
uniqueLength (integer unique_size)	Array words words is not empty	NONE	Return the total number of distinct words with frequency 1 in a text file	unique_size
word_Occurrences (string word , integer count)	Array words is not empty and word not null	word	Return the total number of occurrences of it by Search for certain given word in the LinkedList of array cell indexed with length of that word.	count
particular_length (integer length, integer count)	list words is not empty	length	Searching for the Array words using cell with index equal to length value for all total number of words with a particular length.	count
all_words_occurrences ()	Array words is not empty	NONE	display the words and their occurrences sorted by the total occurrences of each word (from the most frequent to the least), priority queue PQ data will be used to display.	NONE

locations_ occurrences (String word)	Array words is not empty and word not null	word	display the locations of the occurrences of a word starting from the top of the text file, display all information from locations linked list of the word .	NONE
adjacent (String word1, String word2, Boolean flag)	Array words is not empty and word1 and word2 not null	word1, word2	check if two words are available and occurring adjacent to each other in file at least one occurrence of both words is needed to satisfy this operation, by checking if they are neighbors using their locations linked list.	flag
PQSort()	Linked List words is not empty	NONE	Sort all list words according to their occurrences in the file by entering it in PQ list using frequency value as priority and bring it back to words list.	sorted List

d) Provide the time complexity (worst case analysis) for all the operations discussed above using Big O notation. For operations (3) and (4), consider two cases: the first case, when the words in the text file have lengths that are evenly distributed among different lengths (i.e., the words should have different lengths starting from 1 to the longest with k characters), and the second case, when the lengths of words are not evenly distributed. For all operations, assume that the length of the text file is n, the number of unique words is m, and the longest word in the file has a length of k characters.

public int fileLength()	O(1)
public int uniqueLength()	O(1)
public int word_Occurrences(String word)	O(n) (if all words are distributed equally while counting The value will be the same O(1))
public int particular_length(int length)	O(1)
public void_words_occurrences()	O(m)
public void locations_occurrences(String word)	O(n)
public Boolean adjacent (String word1, String word2)	O(n)